ARTINFO
MUSINFO

BULLE TINTE CHNIQ UE#14 UNIVE RSITE PARIS8 INSTIT UTD'IN TELLIG ENCEA RTIFIC IELLE.



Jérôme C H A I L L O U X

O P A L

OPAL est un programme écrit en ALGOL 510 destiné à disposer dans une surface un certain nombre de figures simples (rectangles, losanges, cercles, ellipses).

Cette surface sera assimilée à un ensemble de points ne pouvant prendre que deux états possibles (ex: blanc ou noir).

L'inclusion des figures sera obtenue par changement d'état des points à l'intérieur de ces figures.

I - Introduction des données.

En début de programme les données suivantes seront introduites sur M.A.E. :

- un nombre (NB) destiné à "lancer" la procédure aléatoire,
- le nombre de lignes (IM), de colonnes (JM) de la surface totale,
- le nombre de rectangles (NREC), de losanges (NLOS), de cercles (NCER), et d'ellipses (NELL) à y inclure.

/ PROG 012 021/

II - Calcul des paramètres.

En ALGOL 510, la déclaration d'un tableau de dimension assez grande étant impossible (par manque de place en mémoire), la surface totale ne sera pas représentée en mémoire.

Dans un premier temps, les paramètres de toutes les figures à inclure vont être calculés et stockés dans les tableaux REC, LOS, CER, ELL.

Ces paramètres sont calculés aléatoirement. Une seule règle générale est fixée : l'intersection de deux figures ne peut être
- que vide (figures disjointes),
- qu'un point,
- qu'une surface.

En aucun cas elle ne peut être un segment de droite, ceci afin de préserver la perception des figures.

Les procédures utilisées sont :
- AL, procédure aléatoire classique, pouvant relancer le programme, si les paramètres ne peuvent être trouvés.

  / PROG 100 /
- SURI, SURJ qui indiquent si les abcisses ou les ordonnées des paramètres trouvés ont déjà été utilisées.

  / PROG 200 300 /
- SURLOS qui indique si le paramètre trouvé se situe sur un périmètre de losanges. / PROG 400 /

Pour chaque rectangle on calculera deux abcisses et deux ordonnées, pour chaque losange et pour chaque cercle une abcisse, une ordonnée et un rayon, et pour chaque ellipse une abcisse, une ordonnée, un grand axe et un petit axe. / PROG 500 600 700 800 /

Les paramètres trouvés sont ensuite imprimés sur M.A.E.

III - Sortie de la figure

La surface totale est divisée en blocs de 80 colonnes sur le nombre total de lignes.

Ces blocs seront extraits un à un. La fin de chaque bloc sera signalée par l'impression sur M.A.E. du libellé FB et l'arrêt du calculateur.

/ PROG 900 /

Les blocs peuvent être

- imprimés sur M.A.E. CLE(4), sur 80 caractères, les deux états étants * et espace .

  /FIG 1 /

- perforés CLE(3), sur 80 caractères avec ces mêmes états et adjonction en fin de ligne du caractère Z. La procédure standard PERF rajoutant en fin d'exécution 3 espaces et un RC, le ruban aura comme aspect en fin de ligne :
  / 80 caractères / / 3 espaces RC / / Z / / 3 espaces RC /

Ce ruban a la propriété de pouvoir être lu directement par une FLEXO (CODE CAB) déconnectée. En position "minuscules" les correspondances seront :

* (CAE) ——→ ∞	espace (CAE) ——→ "
RC (CAE) ——→ :	Z (CAE) ——→ RC

/ FIG 2 /

Problèmes posés par la taille des caractères.

Sur M.A.E CAE le rapport hauteur sur longueur d'un caractère étant voisin de 5/3 (H = 0,425 cm ; L = 0,25 cm), il en résulte une transformation de la figure.

Dans la figure 1 "l'ellipse" provient de l'équation d'un cercle et le "cercle" de l'équation d'une ellipse de rapport grand axe sur petit axe voisin de 5/3 (19/11).

Sur FLEXO CAB ce rapport est voisin de 2/1 (H = 0,42 cm ; L = 0,21 cm).

+++++++++++++++++++++++++++++++++++++++
+ TRADUCTEUR RUBAN CAE CAB 500 +
+++++++++++++++++++++++++++++++++++++++

Ce programme écrit en langage externe CAB permet de lire un ruban CAE. Cette lecture ne posera aucun problème car le contrôle d'imparité est le même (le nombre de perforations est impair).

D'autre part il permet de faire correspondre à deux lignes de ruban CAE une ligne sur FLEXO, ce qui conservera les dimensions exactes des figures en ramenant le rapport hauteur sur longueur des caractères voisin de 1/1.

Pour cela on utilisera les caractères
(EXP)° et espace pour la ligne haute
(IND)° et espace pour la ligne basse.

/ FIG 3/

Description du programme

1 Lecture du début du ruban jusqu'à la rencontre du caractère RC(CAB) = Z(CAE) ; ce qui fait perdre la première ligne mais évite de placer manuellement le ruban au début exact de la ligne.
/ 1,4 à 1,6 /

2 Introduction de la ligne haute en piste 2,0, de la ligne basse en piste 3,0 jusqu'à la rencontre du caractère RC(CAB)
/ 1,7 à 1,29 /

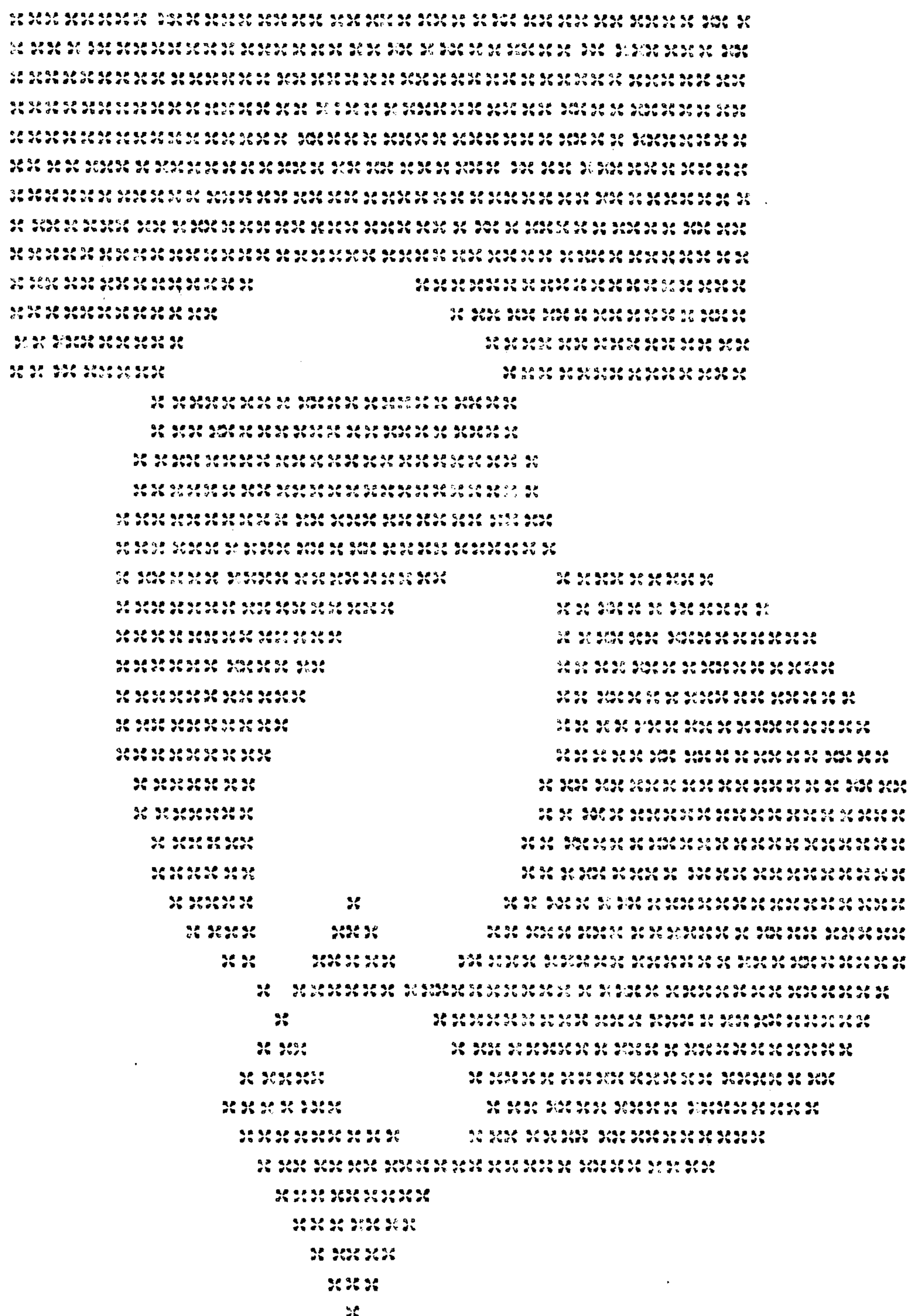
3 Comparaison du nombre de caractères introduits dans les deux lignes. Le programme s'arrête si ces deux nombres sont différents.
/ 1,30 à 1,33 /

4 Recherche de l'adresse du dernier caractère non blanc de chaque ligne. La plus grande est conservée. / 1,35 à 1,44 /

5 Impression
- pour la ligne haute (EXP)° puis espace arrière si * (CAE)
rien si espace (CAE)
- pour la ligne basse (IND)° si * (CAE)
espace si espace (CAE)
jusqu'à l'adresse trouvée en (4) , puis RC et retour en (2).
/ 1,57 à 1,74 /

En position "VARIANTE" allumée, le contraire se produit, créant ainsi une image négative.

OPAL 1



FB

FIG 1

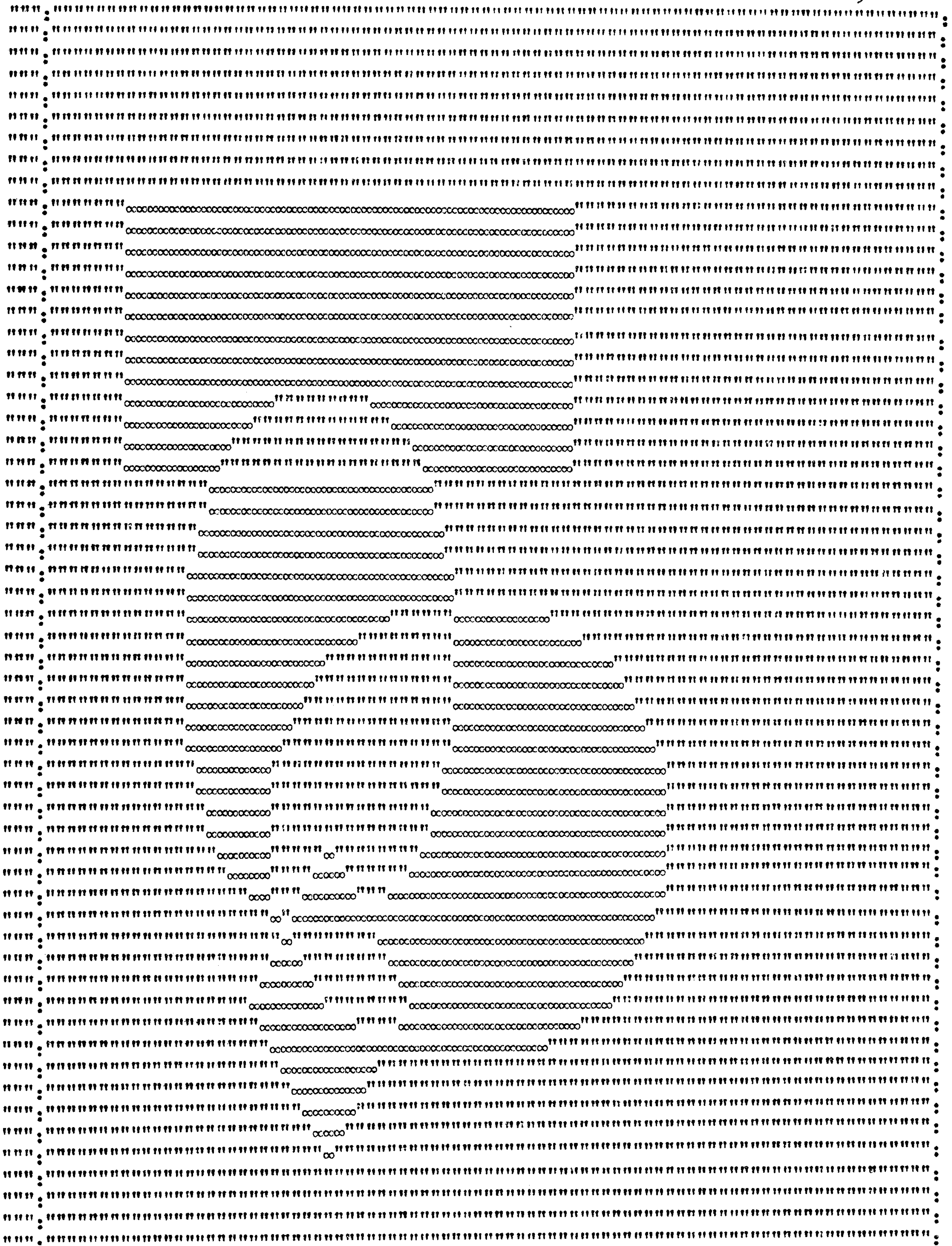


FIG -2-

1,(E

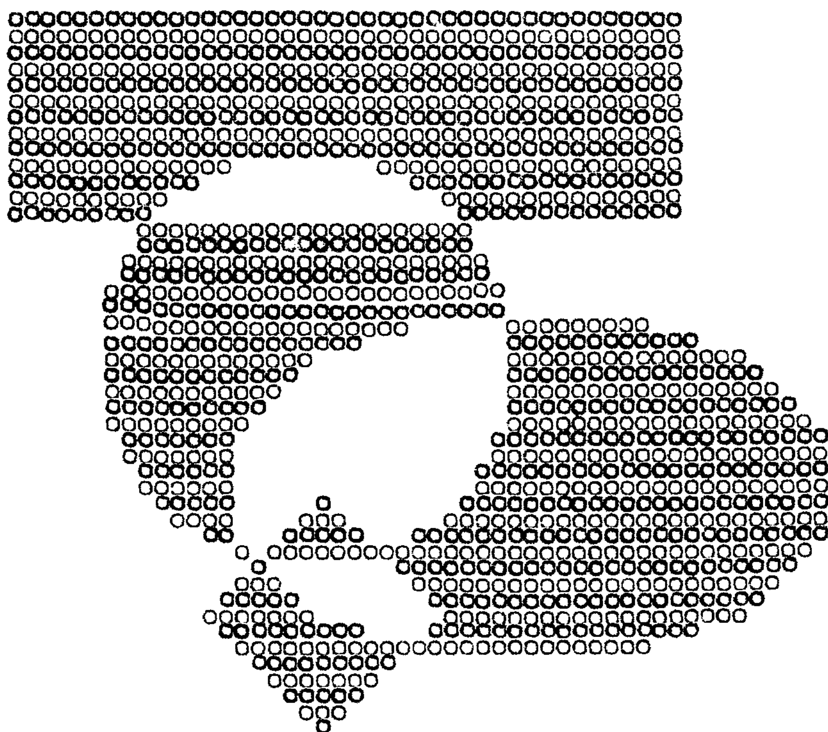


FIG -3-

CAB		RUBAN	CAE	
CØDE INTERNE	HAUT BAS		CØDE RUBAN ØCTAL	HAUT BAS
00	0 E^{0}	-0---•----	20	+ /
01	1 E^{1}	00---•---0	61	
02	2 E^{2}	00---•--0-	62	S
03	3 E^{3}	-0---•--00	23	C
04	4 E^{4}	00---•0--	64	U
05	5 E^{5}	-0---•0-0	25	E
06	6 E^{6}	-0---•00-	26	F
07	7 E^{7}	00---•000	67	X
08	8 E^{8}	00-0•---	70	Y
09	9 E^{9}	-0-0•---0	31	I
10	+ •	-0-0•-0-	32	
11	- I_0	00-0•-00	73	
12	, I_7	-0-0•0--	34	
13	×)	00-0•0-0	75	(,
14	= I_8	00-0•00-	76	TAB
15	ESP	-0-0•000	37	
16	/ (	000-•---	60	
17	Δ I_8	-00-•--0	21	A
18		-00-•-0-	22	B
19		000-•-00	63	T
20		-00-•0--	24	D
21		000-•0-0	55	V
22		000-•00-	56	W
23	MIN	-00-•000	27	G
24	MAJ	-000•---	30	H
25	RC	0000•--0	71	Z
26	E.A.	0000•-0-	72	
27		-000•-00	33	* ;
28		0000•0--	74	
29		-000•0-0	35	
30	TAB	-000•00-	36	MIN
31	EFF.	0000•000	77	EFF.

CAB		RUBAN	CAE	
CØDE INTERNE	HAUT BAS		CØDE RUBAN ØCTAL	HAUT BAS
32	AVB.	0----•----	40	- =
33	A °	-----•---0	01	1 "
34	B I_5	-----•--0-	02	2 &
35	C I_3	0----•-00	43	L
36	D ≤	-----•0--	04	4 >
37	E ≢	0----•0-0	45	N
38	F	0----•00-	46	Ø
39	G ?	-----•000	07	7 ±
40	H <	---0•----	10	8 _
41	I ı	0--0•---0	51	R
42	J ȷ	0--0•-0-	52	
43	K k	---0•-00	13	@
44	L ∞	0--0•0--	54	* ?
45	M ≥	---0•0-0	15	E.A.
46	N I_6	---0•00-	16	MAJ
47	Ø :	0--0•000	57	RC
48	P ≭	--0-•----	00	0 :
49	Q ∗	0-0-•--0	41	J
50	R ↓	0-0-•-0-	42	K
51	S ∠	--0-•-00	03	3 ≤
52	T >	0-0-•0--	44	M
53	U !	--0-•0-0	05	5 %
54	V I_4	--0-•00-	06	6 !
55	W I_1	0-0-•000	47	P
56	X I_2	0-00•----	50	Q
57	Y √	--00•--0	11	9 £
58	Z "	--00•-0-	12	ESP
59		0-00•-00	53	$ ≠
60		--00•0--	14	°
61		0-00•0-0	55	) '
62		0-00•00-	56	NOIR
63		--00•000	17	ROUG

CØDE - RUBAN CAB ↔ CAE

Configuration ruban

CAB	CAE
Parité	C6
C6	C5
C5	Parité
C4	C4
• Entraînement	• Entraînement
C3	C3
C2	C2
C1	C1

Le nombre de trous est impair dans les 2 cas

```
***********
*  OPAL   *
***********

001   'DEB'
002     'ENT'IM,JM,L,NB,NAL,NLOS,NREC,NCER,NELL;

100     'ENT' 'PROC' AL(N) ; 'ENT' N ;
101     'DEB' NB:=5*NB ;
102        'SI' NB 'SUG' 1093 'ALO' NB:=NB-(NB%1093)*1093 ;
103        NAL:=NAL+1 ;
104        'SI' NAL 'INF' 1000 'ALO' 'ALL' FINAL ;
105        EXL(&!#1000_AL@) ; IMPR ; NAL:=0 ;
106        'SI' CLE(7) 'ALO' 'ALL' BEGIN ;
107     FINAL :  AL:=NB-(NB%N)*N+1 ;
108     'FIN' DE LA PROCEDURE ALEATOIRE ;

010     BEGIN :
011     EXL(&!NB_IM_JM@) ; IMPR ;
012     LICLAV(NB,IM,JM) ;
013     NAL:=0;
015     L:= 'SI' IM 'SUP' JM 'ALO' JM%10 'SIN' IM%10 ;

020     EXL(&!NREC_NLOS_NCER_NELL@) ; IMPR ;
021     LICLAV(NREC,NLOS,NCER,NELL) ;

050     'DEB' 'ENT' 'TAB' LOS.(0:NLOS,1:3). , REC.(0:NREC,1:4). ,
051        CER.(0:NCER,1:3). , ELL.(0:NELL,1:4). ;
052        'BOO' 'TAB' IB.(1:IM). , JB.(1:JM). ;
053        'ENT' A,B,C,D,I,J,K,M,T ;
054        'BOO' NC,PN ;

200        'BOO' 'PROC' SURI(N) ; 'ENT' N ;
201        SURI:= IB.(N-2). 'OU' IB.(N-1).
202        'OU' IB.(N). 'OU' IB.(N+1). 'OU' IB.(N+2).;

300        'BOO' 'PROC' SURJ(N) ; 'ENT' N ;
301        SURJ:= JB.(N-2). 'OU' JB.(N-1).
302        'OU' JB.(N). 'OU' JB.(N+1). 'OU' JB.(N+2). ;

400        'BOO' 'PROC' SURLOS(N,I,J) ; 'ENT' N,I,J ;
401        'DEB' 'ENT' LA,LB,LC1,LC2,COMP ;
402           'POU' COMP:=1 'PAS' 1 'JUS' N 'FAI'
403           'DEB' LC1:=LOS.(COMP,3).+2 ; LC2:=LOS.(COMP,3).-2 ;
404              LA:=I-LOS.(COMP,1). ; LB:=J-LOS.(COMP,2). ;
405              'SI' LA+LB 'SUP' LC1 'OU' LA-LB 'SUP' LC1
406              'OU' LB-LA 'SUP' LC1 'OU' -LA-LB 'SUP' LC1
407              'ALO' 'ALL' FALSE ;
408              'SI' LA+LB 'INF' LC2 'ET' LA-LB 'INF' LC2
409              'ET' LB-LA 'INF' LC2 'ET' -LA-LB 'INF' LC2
410              'ALO' 'ALL' FALSE 'SIN' 'ALL' TRUE ;
411           FALSE : 'FIN' ;
412           SURLOS:= 'FAUX' ; 'ALL' FSURLO ;
413           TRUE : SURLOS:= 'VRAI' ;
414        FSURLO : 'FIN' DE SURLOS ;

060        'POU' K:=1 'PAS' 1 'JUS' IM 'FAI' IB.(K).:='FAUX' ;
061        'POU' K:=1 'PAS' 1 'JUS' JM 'FAI' JB.(K).:='FAUX' ;
```

```
'COM'  ** CALCUL DES PARAMETRES DES LOSANGES  ** ;
EXL(&!LOSANGES@) ; EXE(3,NLOS) ; IMPR ;
'POU' K:=1 'PAS' 1 'JUS' NLOS 'FAI'
'DEB' LOSI : C:=AL(L*5-12)+6 ;
   A:=AL(IM-2*C-6)+C+3 ;
   B:=AL(JM-2*C-6)+C+3 ;
   'SI' SURI(A+C) 'OU' SURI(A-C) 'OU' SURI(A)
   'OU' SURJ(B+C) 'OU' SURJ(B-C) 'OU' SURJ(B)
   'ALO' 'ALL' LOSI ;
   IB.(A).:=IB.(A+C).:=IB.(A-C).:=
   JB.(B).:=JB.(B+C).:=JB.(B-C).:= 'VRAI' ;
   LOS.(K,1).:=A ; LOS.(K,2).:=B ; LOS.(K,3).:=C ;
EXE(4,A,B,C) ; IMPR ; 'FIN' LOSANGES ;

'COM'  ** CALCUL DES PARAMETRES DES ELLIPSES  ** ;
EXL(&!ELLIPSES@) ; EXE(3,NELL) ; IMPR ;
'POU' K:=1 'PAS' 1 'JUS' NELL 'FAI'
'DEB' ELLI : C:=AL(IM%2-20)+10 ;
   D:=AL(JM%2-20)+10 ;
   A:=AL(IM-2*(C+5))+C+5 ;
   B:=AL(JM-2*(D+5))+D+5 ;
   'SI' SURI(A+C) 'OU' SURI(A-C) 'OU' SURJ(B+D)
   'OU' SURJ(B+D) 'ALO' 'ALL' ELLI ;
   IB.(A+C).:=IB.(A-C).:=
   JB.(B+D).:=JB.(B-D).:= 'VRAI' ;
   ELL.(K,1).:=A ; ELL.(K,2).:=B ;
   ELL.(K,3).:=C ; ELL.(K,4).:=D ;
EXE(4,A,B,C,D) ; IMPR ; 'FIN' ELLIPSES ;

'COM'  ** CALCUL DES PARAMETRES DES CERCLES  ** ;
EXL(&!CERCLES@) ; EXE(3,NCER) ; IMPR ;
'POU' K:=1 'PAS' 1 'JUS' NCER 'FAI'
'DEB' CERI : C:=AL(L*5-20)+10 ;
   A:=AL(IM-2*(C+5))+C+5 ;
   B:=AL(JM-2*(C+5))+C+5 ;
   'SI' SURI(A+C) 'OU' SURI(A-C) 'OU' SURJ(B+C)
   'OU' SURJ(B-C) 'ALO' 'ALL' CERI ;
   IB.(A+C).:=IB.(A-C).:=
   JB.(B+C).:=JB.(B-C).:= 'VRAI' ;
   CER.(K,1).:=A ; CER.(K,2).:=B ; CER.(K,3).:=C ;
EXE(4,A,B,C) ; IMPR ; 'FIN' CERCLES ;

'COM'  ** CALCUL DES PARAMETRES DES RECTANGLES  ** ;
EXL(&!RECTANGLES@) ; EXE(3,NREC) ; IMPR ;
'POU' K:=1 'PAS' 1 'JUS' NREC 'FAI'
'DEB' INDI : A:=AL(IM-8)+4 ;
   'SI' SURI(A) 'ALO' 'ALL' INDI ; IB.(A).:= 'VRAI' ;
   LONI : B:=AL(IM-8)+4 ;
   'SI' SURI(B) 'ALO' 'ALL' LONI ; IB.(B).:= 'VRAI' ;
   'SI' A 'INF' B 'ALO' 'ALL' INDJ ;
   T:=A ; A:=B ; B:=T ;
   INDJ : C:=AL(JM-8)+4 ;
   'SI' SURJ(C) 'ALO' 'ALL' INDJ ; JB.(C).:= 'VRAI' ;
   LONJ : D:=AL(JM-8)+4 ;
   'SI' SURJ(D) 'ALO' 'ALL' LONJ ; JB.(D).:= 'VRAI' ;
   'SI' C 'INF' D 'ALO' 'ALL' TSL ;
   T:=C ; C:=D ; D:=T ;
   TSL :'ALL' 'SI' SURLOS(NLOS,A,C) 'OU' SURLOS(NLOS,A,D)
   'OU' SURLOS(NLOS,B,C) 'OU' SURLOS(NLOS,B,D)
   'ALO' CHGR 'SIN' LOAD ;
   CHGR : IB.(A).:=IB.(B).:=JB.(C).:=JB.(D).:= 'FAUX' ;
   'ALL' INDI ;
   LOAD : REC.(K,1).:=A ; REC.(K,2).:=B ;
   REC.(K,3).:=C ; REC.(K,4).:=D ;
EXE(4,A,B,C,D) ; IMPR ; 'FIN' RECTANGLES ;
```

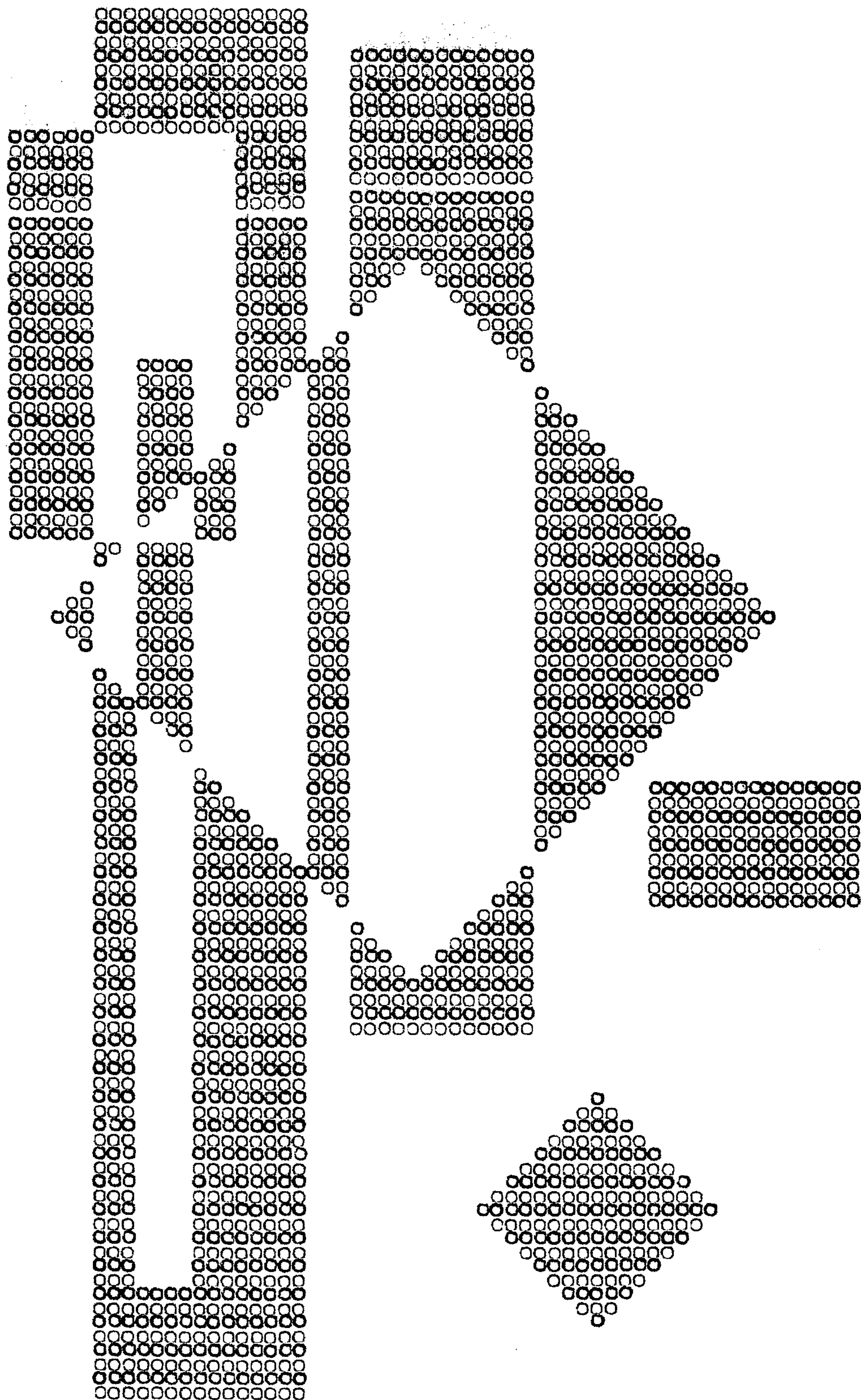
```
900      PAUSE(1) ;
901      'SI' CLE(1) 'ALO' 'ALL' BEGIN ;
902      PN:= 'SI' CLE(6) 'ALO' 'VRAI' 'SIN' 'FAUX' ;
910      'POU' M:=0 'PAS' 1 'JUS' JM%80 'FAI'
911      'DEB' EXL(&!#_OPAL_1!@) ; IMPR ;
912         'POU' I:=1 'PAS' 1 'JUS' IM 'FAI'
913         'DEB' 'POU' J:=1+M*80 'PAS' 1 'JUS' 80*(M+1) 'FAI'
914            'DEB' NC:=PN ;
920               'POU' K:=1 'PAS' 1 'JUS' NLOS 'FAI'
921               'DEB' A:=I-LOS.(K,1). ;
922                  B:=J-LOS.(K,2). ; C:=LOS.(K,3). ;
923                  'SI' A+B 'ING' C 'ET' A-B 'ING' C
924                  'ET' B-A 'ING' C 'ET' -A-B 'ING' C
925               'ALO' NC:= 'NON' NC  'FIN' ;
930               'POU'K:=1 'PAS' 1 'JUS' NREC 'FAI'
931               'SI' I 'SUG' REC.(K,1). 'ET' I 'ING' REC.(K,2).
932               'ET' J 'SUG' REC.(K,3). 'ET' J 'ING' REC.(K,4).
933               'ALO' NC:= 'NON' NC ;
940               'POU' K:=1 'PAS' 1 'JUS' NCER 'FAI'
941               'DEB' A:=I-CER.(K,1). ;
942                  B:=J-CER.(K,2). ; C:=CER.(K,3). ;
943                  'SI' A*A+B*B 'INF' C*C
944               'ALO' NC:= 'NON' NC  'FIN' ;
950               'POU' K:=1 'PAS' 1 'JUS' NELL 'FAI'
951               'DEB' C:=ELL.(K,3). ; D:=ELL.(K,4). ;
952                  A:=I-ELL.(K,1). ; B:=J-ELL.(K,2). ;
953                  'SI' A/C*A/C+B/D*B/D 'INF' 1
954               'ALO' NC:= 'NON' NC  'FIN' ;
959               'SI' NC 'ALO' EXL(&*@) 'SIN' EXL(&_@)
960            'FIN' POINT ;
961            'SI' CLE(4) 'ALO' IMPR ;
962            'SI' CLE(3) 'ALO'
963            'DEB' PERF ; EXL(&Z@) ; PERF 'FIN' ;
964            'SI' CLE(5) 'ALO'
965               'DEB' EXE(4,IM,I) ; IMPR 'FIN' ;
970         'FIN' LIGNE ;
971         EXL(&!!FB@); IMPR ;  PAUSE(5) ;
972      'FIN' BLOC ;
973      EXL(&!!FF@) ; IMPR ;
997      PAUSE(7) ;
998      'ALL' BEGIN ;
999      'FIN' 'FIN' (
```

TRADUCTEUR RUBAN CAE CAB 5()()

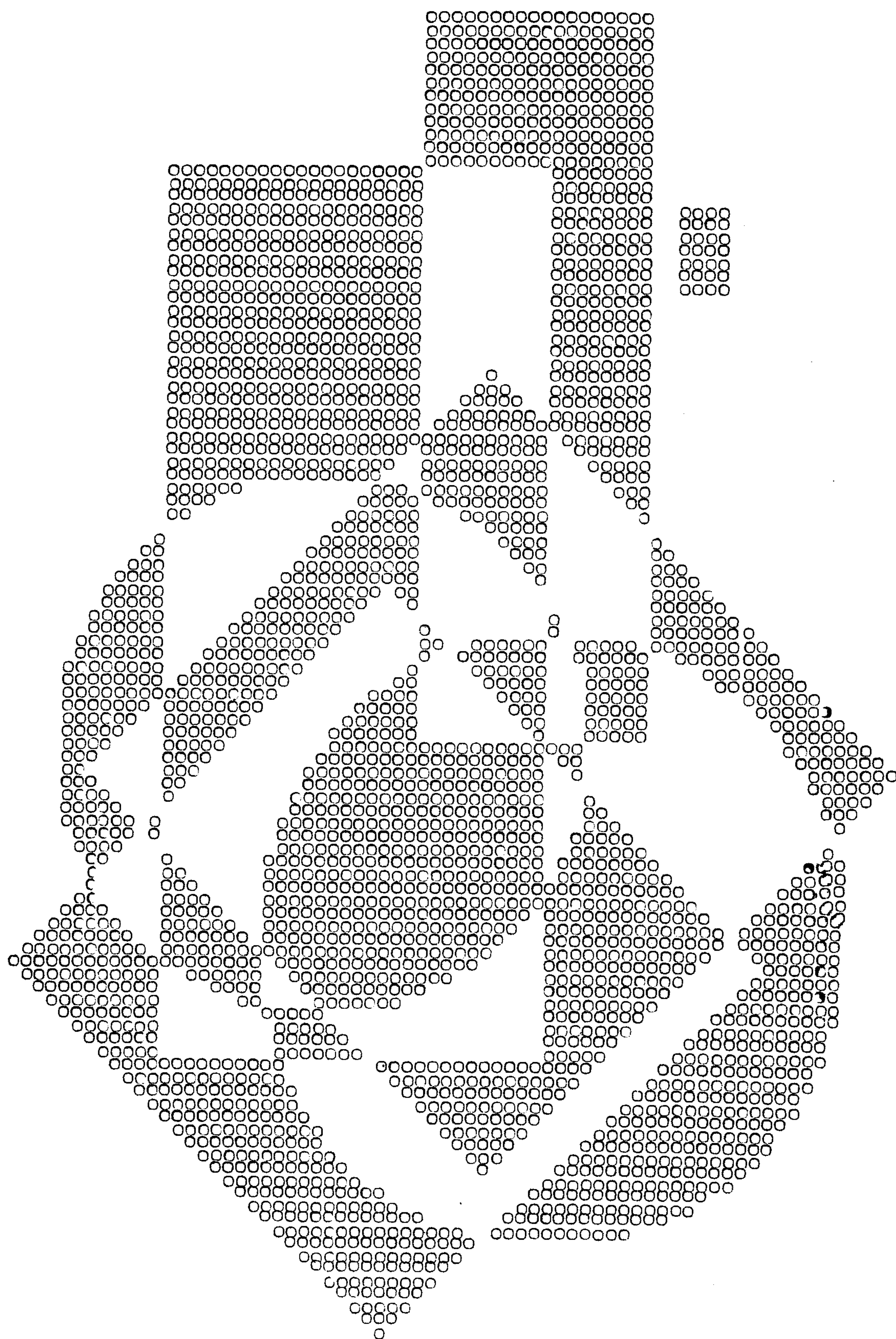
****

1,()M 4()PA		1,4()M 4()PA	
1,()	EV25	1,4()	V1TV44
1,1	EV25	1,41	V1TV58I
1,2	EV23	1,42	RV1,37Y13
1,3	X	1,43	A3ER15
1,4	A1EL()	1,44	A15ER2
1,5	V1TV25	1,45	X
1,6	RV1,4Y13	1,46	S15V1
1,7	A3EV1	1,47	RV1,52Z15
1,8	X	1,48	A1EM3,()+
1,9	A15EV256	1,49	V1TV44
1,1()	S3V1	1,5()	V1TV58I
1,11	A2EV()	1,51	RV1,46Y13
1,12	A1EL()	1,52	A4ER15
1,13	V1TV25	1,53	S15R3
1,14	S2V1Z13	1,54	A5ER4P15
1,15	V1TV58Y13	1,55	A5ER3N15
1,16	V1TV44Y13	1,56	X
1,17	RV1,12Y13	1,57	A15EV()
1,18	T1M()+	1,58	A1EM2,()+
1,19	A15V1	1,59	V1TV44
1,2()	V15TV356Z3	1,6()	V1TV58I
1,21	V15TV484Y3	1,61	EV()Z13
1,22	FZ13	1,62	EV26Z13
1,23	RV1,22Z13	1,63	X
1,24	X	1,64	A1EM3,()+
1,25	RV1,12Z2	1,65	V1TV44
1,26	S15V256Z3	1,66	V1TV58I
1,27	A4ER15Z3	1,67	EV11Z13
1,28	A15EV384Z3	1,68	EV15Y13
1,29	RV1,1()Z3	1,69	V15TR5
1,3()	S15V384	1,7()	A15V1
1,31	V15TR4	1,71	RV1,58Y13
1,32	FY13	1,72	EV25
1,33	RV1,32Y13	1,73	EV23
1,34	X	1,74	RV1,7
1,35	A2ER15	1,75	X
1,36	X	1,76	X
1,37	S15V1	1,77	X
1,38	RV1,43Z15	1,78	X
1,39	A1EM2,()+	1,79	X

1,(E



"



Pierre-Louis NEUMANN

DEFINITION D'UN ESPACE COMPREHENSIBLE

Mon intention dans ce programme est de définir un <u>systeme de construction</u> qui permette de rendre un dessin mentalement <u>intéressant</u>.

Il s'agit de déterminer un espace à deux dimensions pouvant être "compris" par le spectateur après qu'il ait établi les relations entre les éléments de cet espace.

Le système de construction repose sur une <u>règle de construction</u> qui utilise une loi géométrique visuellement identifiable (premier niveau de compréhension : repérage du processus de construction).

Le système est initialisé par un <u>objet de base</u> (choisi par l'utilisateur) qui aura une influence à toutes les étapes de la construction (deuxième niveau de compréhension : repérage du développement spécifique à chaque objet de base)

Le programme CONSTRUCTION 1 permet de construire un espace à deux dimensions à partir d'un objet quelconque, c'est à dire qu'il a pour but de placer un certain nombre d'<u>éléments</u> les uns par rapport aux autres en se référant à un <u>objet de base</u> (ou <u>objet modèle</u>) situé dans le plan.

Chaque élément se présentera sous la forme d'une liste de caractéristiques. D'autre part la position de chacun d'eux sera déterminée par une loi géométrique simple. L'objet modèle sera une liste d'éléments qui devront posséder au moins les mêmes caractéristiques que ceux que l'on propose pour la construction. Cette liste s'élargira en cours de programme par adjonction de chaque nouvel élément positionné.

Le dessin final dépendra donc du choix du nombre d'éléments, de leurs caractéristiques, et de la forme de base.

Le langage LISP 510, particulièrement adapté au traitement des listes et à l'analyse des caractéristiques, sera le langage utilisé.

DESCRIPTION DU PROGRAMME

1 Les éléments

Tous les éléments seront des segments de droite caractérisés par :

1/ une orientation plane : horizontale H , verticale V

2/ Une longueur quelconque L

3/ Une intensité quelconque I

Exemple : (H 5 1) ligne horizontale de longueur 5 et d'intensité 1

Les éléments de l'"objet modèle" auront comme caractéristiques supplémentaires les coordonnées de leurs extrêmités.

Exemple : (H 5 1 (6.3)(6.8))

2 La loi géométrique

Un élément sera positionné par rapport à un autre élément déjà positionné ; pour celà ils devront avoir même longueur, même orientation et des intensités différentes.

Soient X et Y deux éléments. Soit X déjà positionné et e,f,g,h les coordonnées de Y à calculer.

Exemple : X = (H 5 1 (a.b)(c.d))

Y = (H 5 3 e f g h)

Sachant que A = L(X)/2 les coordonnées de Y seront :

1/ si I(Y) supérieur à I(X) et orientation H ou si I(Y) inférieur à I(X) et orientation V alors e = A-a, f = A+b, g = A-c, h = A+d (Figure 1) sinon e = A+a, f = A-b, g = A+c, h = A-d (Figure 2) .

2/ si I(Y) inférieur à I(X) et même orientation H ou même orientation V alors e = A-a, f = A-b, g = A-c, h = A-d (Figure 3) sinon si I(Y) supérieur à I(X) et même orientation H ou même orientation V alors e = A+a, f = A+b, g = A+c, h = A+d (Figure 4) .

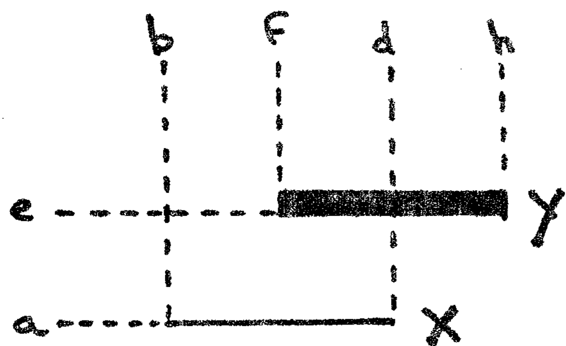
3 Les contraintes

Deux éléments ne devront pas être situés sur une même ligne ou une même colonne.

Deux éléments de même intensité ne devront pas être à la fois parallèles et en regard l'un de l'autre.

Les éléments dont la position respectera les conditions énoncées ci-dessus seront acceptés et mis en queue de la liste "objet-modèle", dans tous les autres cas ils seront refusés. Le listing indiquera les éléments refusés et acceptés, ces derniers suivis de leur position.

Fig 1



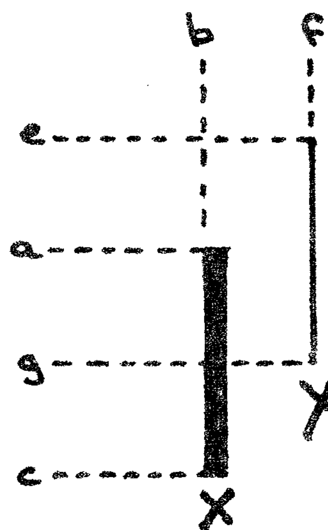
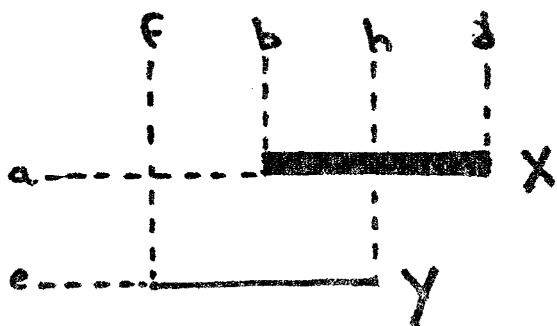


Fig 2



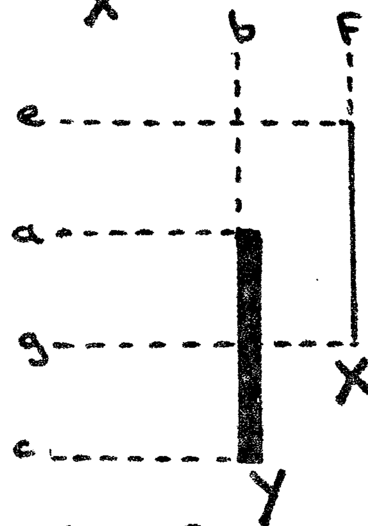
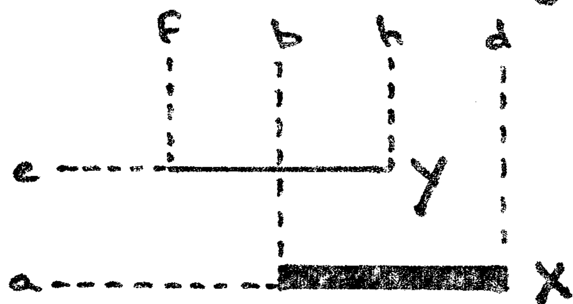


Fig 3



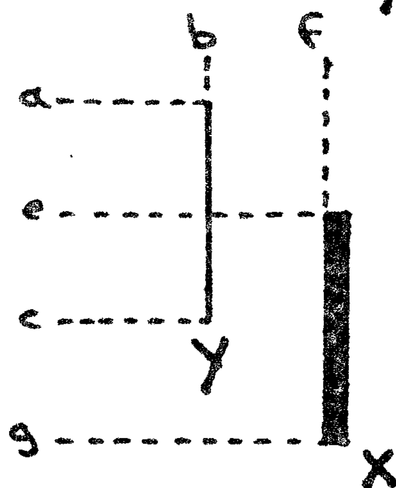
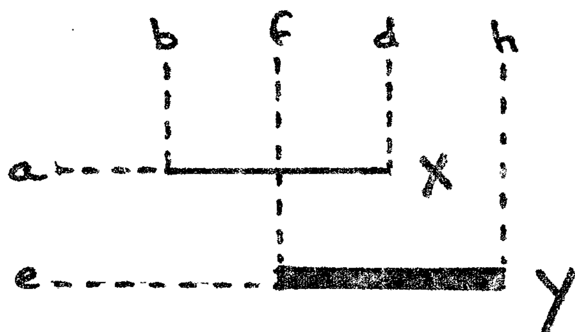
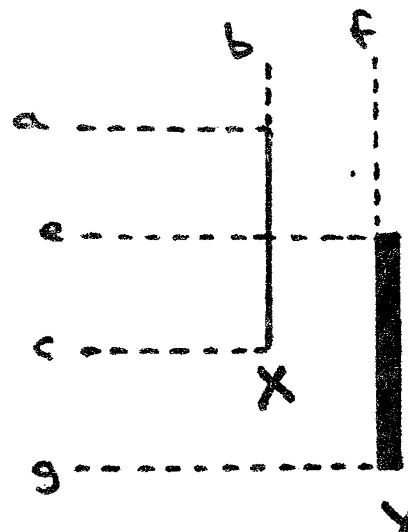


Fig 4





```
*********************
*  CONSTRUCTION 1   *
*********************
```

```
(DE CADDDR (X)
(CADDR(CDR X)))
(DE COTE1 (X)
(CADDR(CDR  X))  )
(DE COTE2 (X)
(COTE1 (CDR X)))
(DE C1A (X) (CAR(COTE1 X)))
(DE C1B (X)(CADR(COTE1 X)))
(DE C2A (X)(CAR(COTE2 X)))
(DE C2B (X) (CADR(COTE2 X)))

(DE APPEND(X Y)
(COND
((NULL X)Y)
(T(CONS(CAR X)
      (APPEND(CDR X) Y))) ))

(DE DELETE(X Y)
(COND
((NULL Y)NIL)
((EQUAL X(CAR Y))
(CDR Y))
(T(CONS(CAR Y)
(DELETE X(CDR Y)))) ))

(DE DIAG (W X Y Z)
                ;**CALCUL DES COORDONNEES**;
(LIST  (X (C1A Z) W)  (Y (C1B Z) W)
    (X (C2A Z) W)  (Y (C2B Z) W)) )

(DE NOBJET (W X Y)
                ;**AJOUTE L ELEMENT POSITIONNE EN QUEUE DE LA
                   LISTE OBJET MODEL**;
(PROGN
(PRINT(SETQ W (APPEND W (LIST
(LIST (CAR X) (CADR X))
(LIST (CADDR X) (CADDDR X))))))
(CONS(APPEND (CAR Y) (LIST W) )
  (DELETE (CADR Y) (CDR Y))) ))

(DE VERIF (V W X Y)
(COND ((EQ W (QUOTE V))
(PLACE V (QUOTE C1B) W X Y (QUOTE CADR) (QUOTE C1A)
                        (QUOTE CAR) ))
(T(PLACE V (QUOTE C1A)  W X Y (QUOTE CAR)
             (QUOTE C1B) (QUOTE CADR)))))
```

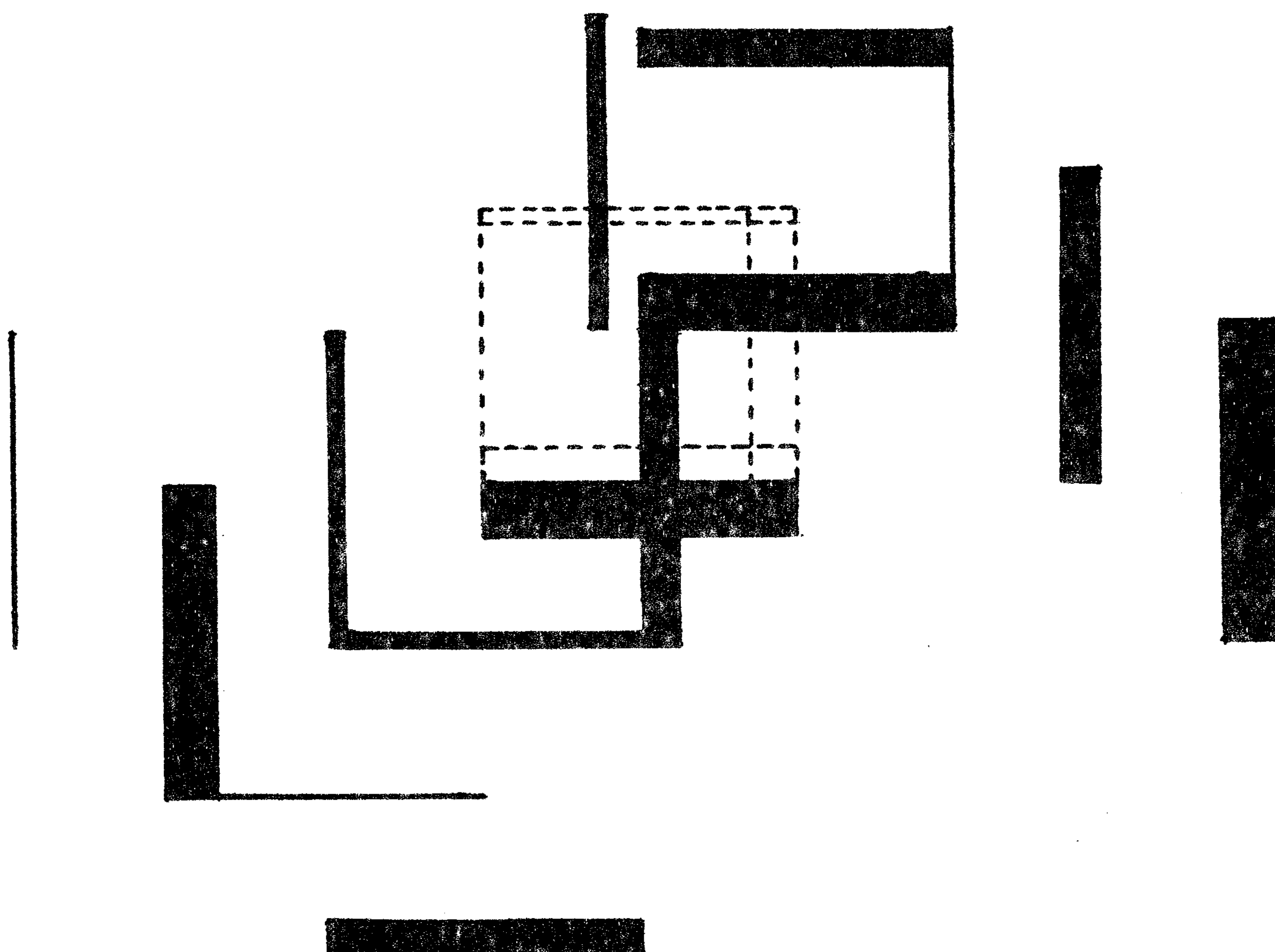
```
(DE POSIT (V X Y Z) (PROG ( O P Q R)
                ;**PROPOSE UNE POSITION DE L ELEMENT EN FONCTION
                   DE SES CARACTERISTIQUES**;
(SETQ P (CADDR Y))
(SETQ R (CADDR Z))
(SETQ O (QUO (CADR Y) 2 ))
(COND ((OR
(AND
(EQ V (QUOTE H))
(GT P R))
(AND
(EQ V (QUOTE V))
(LT P R )) )
(SETQ Q (DIAG O (QUOTE DIFFER) (QUOTE PLUS) Z )))
(T(SETQ Q (DIAG O (QUOTE PLUS) (QUOTE DIFFER) Z ))) )
(AND (VERIF Q V X P ) (RETURN Q))
(COND ((LT P R)
(SETQ Q (DIAG O (QUOTE DIFFER) (QUOTE DIFFER) Z )))
(T(SETQ Q (DIAG O (QUOTE PLUS) (QUOTE PLUS) Z ))) )
(AND (VERIF Q V X P ) (RETURN Q))
(RETURN) ))

(DE OBJET (X)(PROG(O P Q R)
                ;**COMPARE LE PREMIER ELEMENT (A POSITIONNER)
                   ET LA LISTE OBJET MODEL**;
 (SETQ O (CADR X))
(SETQ Q (CAR X))
E (SETQ P (CAR Q))
 (COND ((NULL Q) (RETURN  ))
((AND
 (EQ (CADR P) (CADR O))
(EQ (CAR P) (CAR O))
(NOT(EQ (CADDR P) (CADDR O))))
(GO A)))
D (SETQ Q (CDR Q))
(AND Q (GO E)) (GO F)
A (COND ((EQ (CAR O) (QUOTE H))
(SETQ R(POSIT          (QUOTE H)        (CAR X) O P)))
(T(SETQ R(POSIT          (QUOTE V)        (CAR X) O P))))
(AND (NOT R) (GO D))
(SETQ X (NOBJET O R X))
(GO RE)
F (PRIN1 (QUOTE REFUSE))
(PRINT O)
(SETQ X (DELETE O X))   (GO RE) ))

(DE PLACE (R S U V W X Y Z BID)
                ;**VERIFIE QUE L ELEMENT POSITIONNE
                   SATISFAIT AUX CONTRAINTES**;
(COND((NULL V) T)
((OR
(AND
(EQ (CAAR V) U)
(EQ ( S (SETQ BID (CAR V))) (X R)))
(AND
(EQ (CAAR V) U)
(EQ W (CADDR BID))
(EQ (Y BID) (Z R)) ) ))
(T(PLACE R S U (CDR V) W X Y Z BID))))
```

```
(CPJET(QUOTE
 ((
 (H 6 2 (101 101) (101 106))
 (H 6 3 (106 101) (106 106))
 (V 6 1 (101 101) (106 101))
 (V 6 4 (101 106) (106 106))
 )

 (H 6 1) (H 6 2) (H 6 3) (H 6 4)
 (H 6 1) (H 6 2) (H 6 3) (H 6 4)
 (V 6 1) (V 6 2) (V 6 3) (V 6 4)
 (V 6 1) (V 6 2) (V 6 3) (V 6 4)
 )))
( H 6 1 ( 104 98 ) ( 104 103 ) )
( H 6 2 ( 109 98 ) ( 109 103 ) )
( H 6 3 ( 98 104 ) ( 98 109 ) )
( H 6 4 ( 103 104 ) ( 103 109 ) )
( H 6 1 ( 112 95 ) ( 112 100 ) )
REFUSE ( H 6 2 )
( H 6 3 ( 115 98 ) ( 115 103 ) )
( H 6 4 ( 107 101 ) ( 107 106 ) )
( V 6 1 ( 98 109 ) ( 103 109 ) )
( V 6 2 ( 104 98 ) ( 109 98 ) )
( V 6 3 ( 104 104 ) ( 109 104 ) )
( V 6 4 ( 107 95 ) ( 112 95 ) )
( V 6 1 ( 104 92 ) ( 109 92 ) )
( V 6 2 ( 98 103 ) ( 103 103 ) )
( V 6 3 ( 101 112 ) ( 106 112 ) )
( V 6 4 ( 104 115 ) ( 109 115 ) )
NIL
```

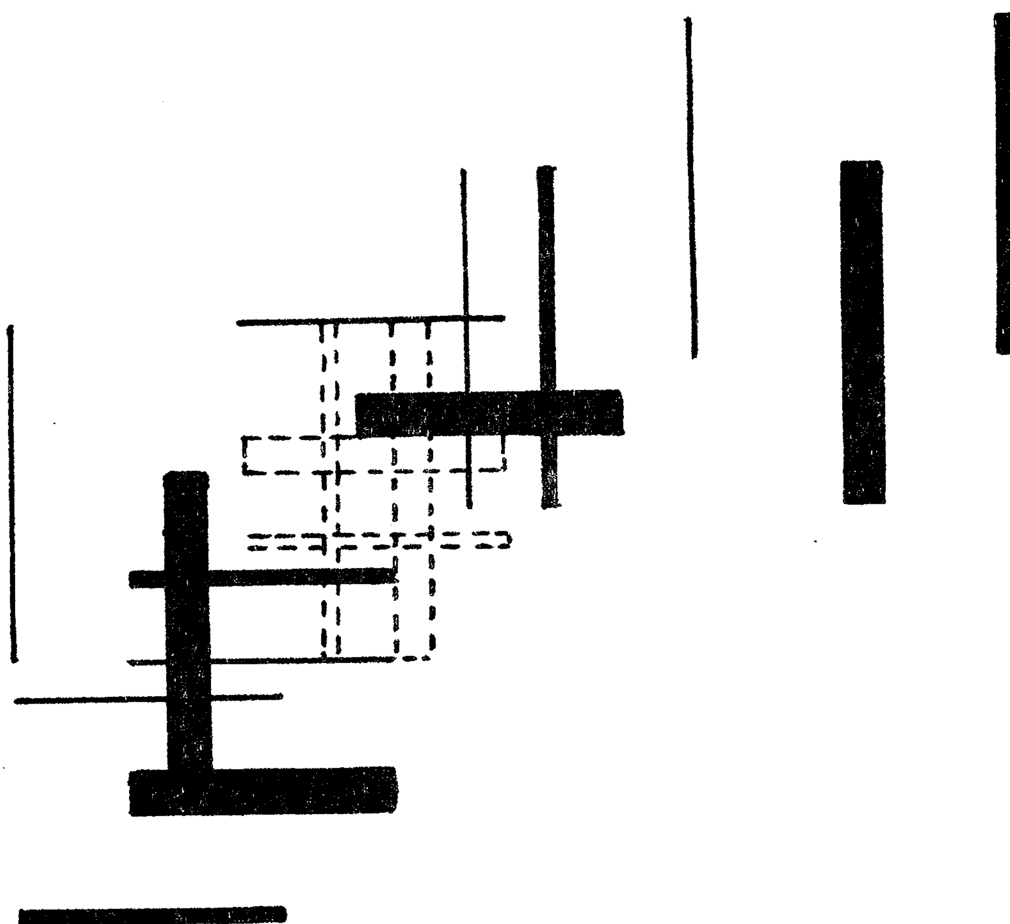


```
(OBJET(QUOTE
 ((
 (H 6 2 (92 98) (92 103))
 (H 5 2 (99 99) (99 103))
 (H 4 2 (111 100) (111 103))
 )

(H 6 3) (H 6 4) (H 6 1) (H 6 0)
(H 5 1) (H 5 0) (H 5 4) (H 5 3)
(H 4 3) (H 4 1) (H  4 0) (H 4 4)
 )))
( H 6 3 ( 89 101 ) ( 89 106 ) )
( H 6 4 ( 95 101 ) ( 95 106 ) )
( H 6 1 ( 86 98 ) ( 86 103 ) )
( H 6 0 ( 98 98 ) ( 98 103 ) )
( H 5 1 ( 101 97 ) ( 101 101 ) )
( H 5 0 ( 97 97 ) ( 97 101 ) )
( H 5 4 ( 103 99 ) ( 103 103 ) )
( H 5 3 ( 105 97 ) ( 105 101 ) )
( H 4 3 ( 109 102 ) ( 109 105 ) )
( H 4 1 ( 107 100 ) ( 107 103 ) )
REFUSE ( H 4 0 )
( H 4 4 ( 113 102 ) ( 113 105 ) )
NIL
```

```
(CPUET(QUOTE
 ((
 (V 9 2 (101 103) (109 103))
 (V 9 3 (101 105) (109 105))
 (H 7 2 (106 101) (106 107))
 (H 7 3 (104 101) (104 107))
 )

 (H 7 1) (H 7 2) (H 7 3)
 (H 7 1) (H 7 2) (H 7 3)
 (H 7 1) (H 7 2) (H 7 3)
 (V 9 1) (V 9 2) (V 9 3)
 (V 9 1) (V 9 2) (V 9 3)
 (V 9 1) (V 9 2) (V 9 3)
 )))
( H 7 1 ( 109 98 ) ( 109 104 ) )
( H 7 2 ( 107 98 ) ( 107 104 ) )
( H 7 3 ( 103 104 ) ( 103 110 ) )
( H 7 1 ( 110 95 ) ( 110 101 ) )
REFUSE ( H 7 2 )
( H 7 3 ( 113 98 ) ( 113 104 ) )
( H 7 1 ( 100 101 ) ( 100 107 ) )
( H 7 2 ( 116 95 ) ( 116 101 ) )
REFUSE ( H 7 3 )
( V 9 1 ( 97 107 ) ( 105 107 ) )
( V 9 2 ( 97 109 ) ( 105 109 ) )
( V 9 3 ( 105 99 ) ( 113 99 ) )
( V 9 1 ( 93 113 ) ( 101 113 ) )
REFUSE ( V 9 2 )
( V 9 3 ( 97 117 ) ( 105 117 ) )
( V 9 1 ( 101 95 ) ( 109 95 ) )
( V 9 2 ( 93 121 ) ( 101 121 ) )
REFUSE ( V 9 3 )
NIL
```



A V E R T I S S E M E N T

Le présent bulletin répond à une visée toute didactique : livrer sous forme accessible aux nouveaux venus dans les groupes de travail courants :

- de l'information technique et bibliographique en rapport avec leurs disciplines ;
- des programmes commentés de tous niveaux permettant un accès relativement rapide à des techniques de programmation appropiées, ainsi qu'une implémentation aisée.

On s'est efforcé, dans la mesure du possible, de ne pas établir de clivage trop net entre les disciplines artistiques et scientifiques concernées (musique, arts plastiques, poésie, architecture, logique, informatique), mais tout au contraire, ne serait-ce que par des techniques de programmation communes.

L'aspect pédagogique du présent bulletin reflète une préoccupation constante du groupe, à savoir ne pas se satisfaire en dernier ressort de méthodes de programmation trop élémentaires.

Le contenu des **textes** et des programmes n'engage que leurs auteurs.

Pour tous renseignements et composition des livraisons à venir, s'adresser à Jacques ARVEILLER, Département d'Informatique, Université PARIS VIII, Route de la Tourelle, Paris XII°.

Pour tout envoi, s'adresser à Patrick GREUSSAY, même adresse.

+=+=+=+=+=+=+